Aufgabe 1 (Reguläre Ausdrücke und endliche Automaten)

(a)

Im folgenden bezeichnen wir den regulären Audruck, der die Sprache L beschreibt mit $reg(L)$.

(i) $reg(L_1) = (\mathbf{a}^*\mathbf{b} \mid \mathbf{b} \mid \mathbf{c})^* \, \mathbf{a}^*$

(ii) $reg(L_2) = (\mathbf{b} \mid \mathbf{c} \mid (\mathbf{a}\, (\mathbf{b} \mid \mathbf{c})^* \, \mathbf{a}\, (\mathbf{b} \mid \mathbf{c})^* \, \mathbf{a}))^*$

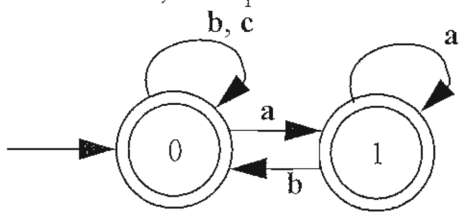
(iii) $reg(L_3) = (\mathbf{aa} \mid \mathbf{b} \mid \mathbf{c})^*$

(iv) $reg(L_4) = (\mathbf{b} \mid \mathbf{c})^* \, (\mathbf{a} \mid \mathbf{b})^*$

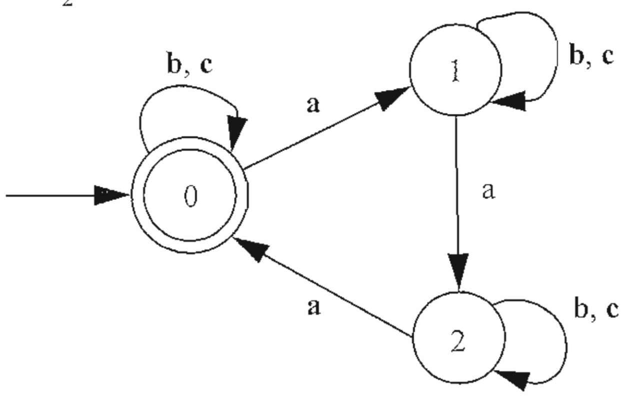
(v) $reg(L_5) = ((\mathbf{a}^*\mathbf{c}) \mid (\mathbf{c} \mid \mathbf{b}))^*$

(b)

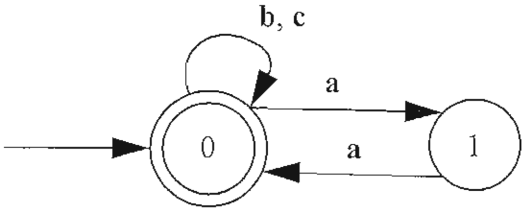
Ein Automat, der L_1 erkennt ist:



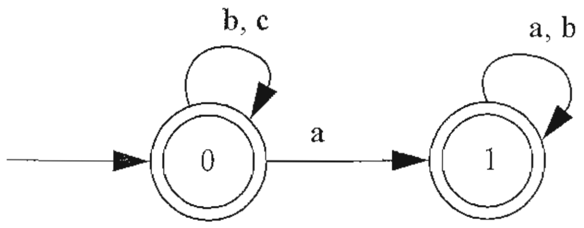
Ein L_2 erkennender Automat ist:



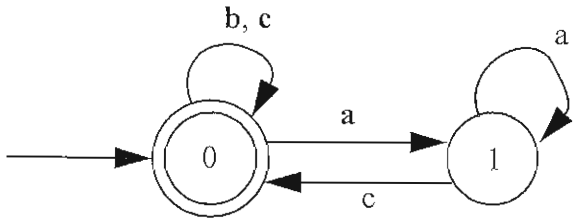
L_3 wird durch folgenden Automaten erkannt:



Der folgende Automat erkennt L_4:



Die Sprache L_5 kann durch folgenden Automaten erkannt werden:



Aufgabe 2

(a)

Die FIRST_1-Mengen sind:

$\mathrm{FIRST}(A) = \{\mathbf{i}\}$

$\mathrm{FIRST}(L) = \{\mathbf{t}, \varepsilon\}$

$\mathrm{FIRST}(S) = \{\mathbf{s}\}$

$\mathrm{FIRST}(B) = \{\mathbf{i}\}$

$\mathrm{FIRST}(R) = \{\mathbf{i}\}$

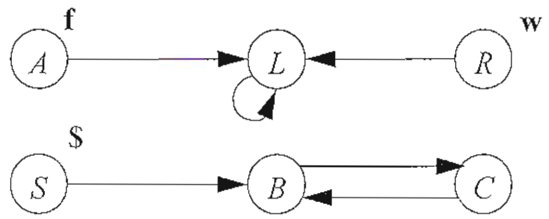
$\mathrm{FIRST}(C) = \{\mathbf{a}, \mathbf{o}, \varepsilon\}$

Damit ergeben sich die folgenden initialen Steuermengen:

1:	$S \to \mathrm{s}A\mathrm{f}R\mathbf{w}B$	$\{\mathbf{s}\}$
2:	$A \to \mathbf{i}\,L$	$\{\mathbf{i}\}$
3:	$R \to \mathbf{i}L$	$\{\mathbf{i}\}$
4:	$L \to \mathbf{ti}L$	$\{\mathbf{t}\}$
5:	$L \to \varepsilon$	$\{\varepsilon\}$
6:	$B \to \mathbf{ici}C$	$\{\mathbf{i}\}$
7:	$C \to \mathbf{a}B$	$\{\mathbf{a}\}$
8:	$C \to \mathbf{o}B$	$\{\mathbf{o}\}$
9:	$C \to \varepsilon$	$\{\varepsilon\}$

(b)

Wir erhalten den folgenden Graphen:



Durch Propagieren der Knotenmarkierungen erhalten wir die folgenden FOLLOW-Mengen:

$\mathrm{FOLLOW}(A) = \{\mathbf{f}\}$

FOLLOW(L) = {**f**, **w**}
FOLLOW(B) = {**$**}
FOLLOW(C) = {**$**}
FOLLOW(R) = {**w**}
FOLLOW(S) = {**$**}

(c)

Unter Verwendung der Nummerierung der Produktionen aus der Lösung zum Aufgabenteil (a) erhalten wir die folgende Parsetabelle:

	s	**f**	**w**	**i**	**t**	**c**	**a**	**o**	**$**
S	1								
A				2					
R				3					
L		5	5		4				
B				6					
C							7	8	9

Aufgabe 3

Die FIRST_1-Mengen sind:

$\text{FIRST}(A) = \{\mathbf{s}\}$

$\text{FIRST}(B) = \{\mathbf{s}\}$

$\text{FIRST}(C) = \{\mathbf{v}\}$

$\text{FIRST}(D) = \{\mathbf{z}\}$

$\text{FIRST}(E) = \{\mathbf{z}\}$

Als kanonische $LR(1)$-Kollektion ergibt sich:

0	Startzustand	$A \to .\, B$ $B \to .\, \mathbf{s}C\mathbf{t}C\mathbf{u}D$	$\{\$\}$ $\{\$\}$
1	$0 \xrightarrow{B} 1$	$A \to B\, .$	$\{\$\}$
2	$0 \xrightarrow{\mathbf{s}} 2$	$B \to \mathbf{s}\, .\, C\mathbf{t}C\mathbf{u}D$ $C \to .\, \mathbf{v}$ $C \to .\, C\mathbf{wv}$	$\{\$\}$ $\{\mathbf{t}, \mathbf{w}\}$ $\{\mathbf{t}, \mathbf{w}\}$
3	$2 \xrightarrow{C} 3$	$B \to \mathbf{s}C\, .\, \mathbf{t}C\mathbf{u}D$ $C \to C\, .\, \mathbf{wv}$	$\{\$\}$ $\{\mathbf{t}, \mathbf{w}\}$
4	$2 \xrightarrow{\mathbf{v}} 4$	$C \to \mathbf{v}\, .$	$\{\mathbf{t}, \mathbf{w}\}$
5	$3 \xrightarrow{\mathbf{t}} 5$	$B \to \mathbf{s}C\mathbf{t}\, .\, C\mathbf{u}D$ $C \to .\, \mathbf{v}$ $C \to .\, C\mathbf{wv}$	$\{\$\}$ $\{\mathbf{u}, \mathbf{w}\}$ $\{\mathbf{u}, \mathbf{w}\}$
6	$3 \xrightarrow{\mathbf{w}} 6$	$C \to C\mathbf{w}\, .\, \mathbf{v}$	$\{\mathbf{t}, \mathbf{w}\}$
7	$5 \xrightarrow{C} 7$	$B \to \mathbf{s}C\mathbf{t}C\, .\, \mathbf{u}D$ $C \to C\, .\, \mathbf{wv}$	$\{\$\}$ $\{\mathbf{u}, \mathbf{w}\}$
8	$5 \xrightarrow{\mathbf{v}} 8$	$C \to \mathbf{v}\, .$	$\{\mathbf{u}, \mathbf{w}\}$
9	$6 \xrightarrow{\mathbf{v}} 9$	$C \to C\mathbf{wv}\, .$	$\{\mathbf{t}, \mathbf{w}\}$
10	$7 \xrightarrow{\mathbf{u}} 10$	$B \to \mathbf{s}C\mathbf{t}C\mathbf{u}\, .\, D$ $D \to .\, D\mathbf{x}E$ $D \to .\, D\mathbf{y}E$ $D \to .\, E$ $E \to .\, \mathbf{z}$	$\{\$\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$

11	$7 \xrightarrow{\mathbf{w}} 11$	$C \to C\mathbf{w}\,.\,\mathbf{v}$	$\{\mathbf{u}, \mathbf{w}\}$
12	$10 \xrightarrow{D} 12$	$B \to \mathbf{s}C\,\mathbf{t}C\mathbf{u}D\,.$ $D \to D\,.\,\mathbf{x}E$ $D \to D\,.\,\mathbf{y}E$	$\{\$\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$
13	$10 \xrightarrow{E} 13$	$D \to E\,.$	$\{\$, \mathbf{x}, \mathbf{y}\}$
14	$10, 16, 17 \xrightarrow{\mathbf{z}} 14$	$E \to \mathbf{z}\,.$	$\{\$, \mathbf{x}, \mathbf{y}\}$
15	$11 \xrightarrow{\mathbf{v}} 15$	$C \to C\mathbf{wv}\,.$	$\{\mathbf{u}, \mathbf{w}\}$
16	$12 \xrightarrow{\mathbf{x}} 16$	$D \to D\mathbf{x}\,.\,E$ $E \to .\,\mathbf{z}$	$\{\$, \mathbf{x}, \mathbf{y}\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$
17	$12 \xrightarrow{\mathbf{y}} 17$	$D \to D\mathbf{y}\,.\,E$ $E \to .\,\mathbf{z}$	$\{\$, \mathbf{x}, \mathbf{y}\}$ $\{\$, \mathbf{x}, \mathbf{y}\}$
18	$16 \xrightarrow{E} 18$	$D \to D\mathbf{x}E\,.$	$\{\$, \mathbf{x}, \mathbf{y}\}$
19	$17 \xrightarrow{E} 19$	$D \to D\mathbf{y}E\,.$	$\{\$, \mathbf{x}, \mathbf{y}\}$

Nummerierung der Produktionen:

1	$A \to B$
2	$B \to \mathbf{s}C\mathbf{t}C\mathbf{u}D$
3	$C \to C\mathbf{wv}$
4	$C \to \mathbf{v}$
5	$D \to D\mathbf{x}E$
6	$D \to D\mathbf{y}E$
7	$D \to E$
8	$E \to \mathbf{z}$

Für die Steuertabelle ergibt sich:

	s	**t**	**u**	**v**	**w**	**x**	**y**	**z**	**$**	*A*	*B*	*C*	*D*	*E*
0	s2										1			
1									acc					
2				s4								3		
3		s5			s6									
4		r4			r4									
5				s8								7		
6				s9										
7			s10		s11									
8			r4		r4									
9		r3			r3									
10								s14					12	13
11				s15										
12						s16	s17		r2					
13						r7	r7		r7					
14						r8	r8		r8					
15			r3		r3									
16								s14						18
17								s14						19
18						r5	r5		r5					
19						r6	r6		r6					

Aufgabe 4

Eine LL(1)-Grammatik, die die Syntax der SQL-Abfrage analysiert ist:

G_{LL} = ({*alist*, *list*, *boolexpr*, *C*, *A*}, {**SELECT**, **FROM**, **WHERE**, **ID**,**,** **COMP**, **AND**, **OR**}, *P*, *A*) mit

P = { *A* → **SELECT** *alist* **FROM** *alist* **WHERE** *boolexpr*

alist → **ID** *list*

list → **,ID** *list*

| ε

boolexpr → **ID COMP ID** *C*

C → **AND** *boolexpr*

| **OR** *boolexpr*

| ε}

Eine LR(1)-Grammatik, die dasselbe leistet ist:

G_{LR} = ({*list*, *boolexpr*, *A*, *C*, *S*}, {**SELECT**, **FROM**, **WHERE**, **ID**,**,** **COMP**, **AND**, **OR**}, *P'*, *S*) mit

P' = { *S* → *A*

A → **SELECT** *list* **FROM** *list* **WHERE** *boolexpr*

list → **ID**

| *list* **,ID**

boolexpr → *C*

| *boolexpr* **AND** *C*

| *boolexpr* **OR** *C*

C → **ID COMP ID**}

Aufgabe 5 (Syntaxgesteuerte Definition)

(a)

Die Grammatik *G*= (*N*, Σ, *P*, *S*) erkennt die in der Aufgabenstellung beschriebene Sprache. Dabei sind:

N = {*S*, *expr*}

Σ = {**id**, **:=**, **num**, **;**, **+** , **-**, *****, **/**, **<**, **=** , **>**, **and**, **or**, **not** }

P = { *S* → **id :=** *expr*;

$expr \rightarrow$ *expr expr* +
| *expr expr* -
| *expr expr* <
| *expr expr* =
| *expr expr* **and**
| *expr expr* **or**
| *expr* **not**
| **num**

}

(b)

Die Einführung weiterer Attribute ist für Elemente aus Σ nicht notwendig. Insgesamt erhalten wir:

A(**id**) = {realval, boolval, error}

A(**num**) = {val}

A(*expr*) = {realval, boolval, error }

Die Attributmengen aller anderen Symbole sind leer. Nun erweitern wir G um semantische Regeln, die die notwendigen Berechnungen durchführen:

$S \rightarrow$ **id** := *expr*;

```
{ if expr.type != error then
    id.type := expr.type;
    if expr.type = bool then
      id.boolval := expr.boolval;
    else
      id.realval := expr.realval;
    end
  else
    id.type = error;
  end
}
```

expr $\rightarrow$ **num**

```
{ expr.type := real;
  expr.realval := num.val;
}
```

$expr_1 \rightarrow expr_2$ **not**

```
{ if exp2.type = bool then
```

$expr_1.type$:= bool;
$expr_1.boolval$:= **not** $expr_2.boolval$;
else
$expr_1.type$:= error;
end
}

$expr_1 \rightarrow expr_2\ expr_3$ +

{ **if** $expr_2.type$ = real **and** $expr_3.type$ = real **then**
$expr_1.type$:= real;
$expr_1.realval := expr_2.realval + expr_3.realval$;
else
$expr_1.type$:= error;
end
}

$expr_1 \rightarrow expr_2\ expr_3$ -

{ **if** $expr_2.type$ = real **and** $expr_3.type$ = real **then**
$expr_1.type := real$;
$expr_1.realval := expr_2.realval - expr_3.realval$;
else
$expr_1.type$:= error;
end
}

$expr_1 \rightarrow expr_2\ expr_3 >$

{ **if** $expr_2.type$ = real **and** $expr_3.type$ = real **then**
$expr_1.type$:= bool;
$expr_1.boolval := expr_2.realval > expr_3.realval$;
else
$expr_1.type$:= error;
end
}

$expr_1 \rightarrow expr_2\ expr_3 =$

{ **if** $expr_2.type$ = real **and** $expr_3.type$ = real **then**
$expr_1.type$:= bool;
$expr_1.boolval := expr_2.realval = expr_3.realval$;
else
$expr_1.type$:= error;
end
}

$expr_1 \rightarrow expr_2\ expr_3$ **and**

{ **if** $expr_2.type$ = bool **and** $expr_3.type$ = bool **then**
  $expr_1.type$:= bool;
  $expr_1.boolval := expr_2.boolval$ **and** $expr_3.boolval$;
else
  $expr_1.type$:= error;
end
}

$expr_1 \rightarrow expr_2\ expr_3$ **or**

{ **if** $expr_2.type$ = bool **and** $expr_3.type$ = bool **then**
  $expr_1.type$:= bool;
  $expr_1.boolval := expr_2.boolval$ **or** $expr_3.boolval$;
else
  $expr_1.type := error$;
end
}

(c)

Die Grammatik enthält nur synthetisierte Attribute.

Aufgabe 6

Der Code für die Funktion *fib* lautet:

```
(1) if n = 0 goto 13
(2) if n = 1 goto 15
(3) n1 := n - 1
(4) n2 := n - 2
(5) valparam n1
(6) call 1
(7) getresult f1
(8) valparam n2
(9) call 1
(10) getresult f2
(11) t := f1 + f2
(12) freturn t
(13) t := 0
(14) freturn t
(15) t := 1
(16) freturn t
```

Der Code der Prozedur *main* lautet dann:

```
(100) y := 7
(101) valparam y
(102) call 1
(103) getresult x
```

Aufgabe 7

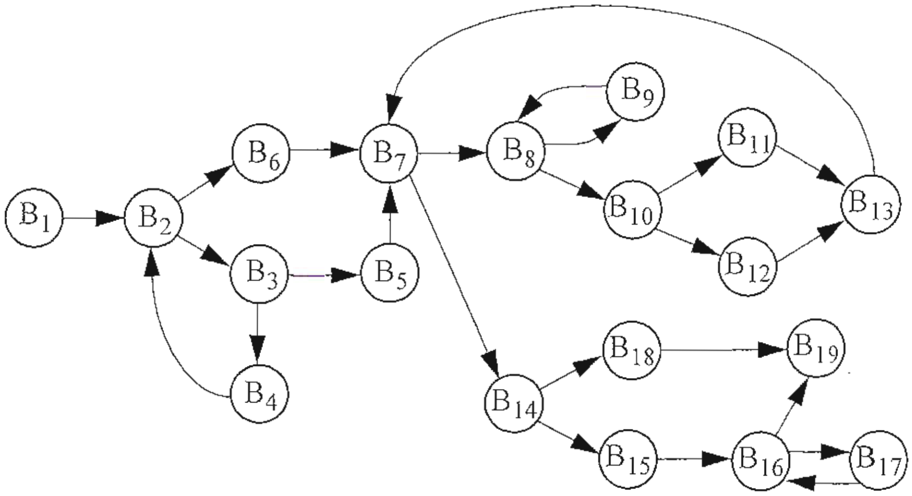
(a)

Es ergibt sich die folgende Einteilung in Basisblöcke:

Block	Begründung für Blockanfang	Zeile	3AC-Anweisung
B_1	Beginn der Prozedur	(1)	`T3 := 0`
B_2	Sprungziel aus (9)	(2)	`T5 := T2 mod 2`
		(3)	`T6 := T1 mod 2`
		(4)	`if T5 = 1 goto 12`
B_3	Nach bedingtem Sprung	(5)	`if T6 = 1 goto 10`
B_4	Nach bedingtem Sprung	(6)	`T1 := T1 / 2`
		(7)	`T2 := T2 / 2`
		(8)	`T3 := T3 + 1`
		(9)	`goto 2`
B_5	Sprungziel aus (5)	(10)	`T4 := -T2`
		(11)	`goto 13`
B_6	Sprungziel aus (4)	(12)	`T4 := T1`
B_7	Sprungziel aus (11), (23)	(13)	`if T4 = 0 goto 24`
B_8	Sprungziel aus (17)	(14)	`T6 := T4 mod 2`
		(15)	`if T6 = 1 goto 18`
B_9	Nach bedingtem Sprung	(16)	`T4 := T4 / 2`
		(17)	`goto 14`
B_{10}	Sprungziel aus (15)	(18)	`if T4 > 0 goto 21`
B_{11}	Nach bedingtem Sprung	(19)	`T2 := -T4`
		(20)	`goto 22`
B_{12}	Sprungziel aus (18)	(21)	`T1 := T4`
B_{13}	Sprungziel aus (20)	(22)	`T4 := T1 - T2`
		(23)	`goto 13`
B_{14}	Sprungziel aus (13)	(24)	`if T3 = 0 goto 30`
B_{15}	Nach bedingtem Sprung	(25)	`T2 := T1`
B_{16}	Sprungziel aus(29)	(26)	`if T3 < 2 goto 31`
B_{17}	Nach bedingtem Sprung	(27)	`T2 := T2 * T1`
		(28)	`T3 := T3 - 1`
		(29)	`goto 26`
B_{18}	Sprungziel aus (24)	(30)	`T2 := 1`
B_{19}	Sprungziel aus (26)	(31)	`return T2`
	Prozedurende		

(b)

Somit ergibt sich der folgende Flussgraph:



(c)

Man unterscheidet bei der Datenflussanalyse danach, ob sie im Flussgraphen *vorwärts* oder *rückwärts* gerichtet ist. Weiterhin unterscheidet man in beiden Gruppen, ob während der Analyse *alle* Böcke oder nur jeweils *ein* benachbarter Block berücksichtigt wird:

	any	*all*
forward	*UD-Verkettung* zur Angabe aller Definitionen, die eine Variablenmenge erreichen	Berechnung der verfügbaren Ausdrücke (zur Eliminierung redundanter Berechnungen)
backward	Ermittlung lebender und toter Variablen	Berechnung der *very busy expressions*

(d)

Bei der Berechnung der vielbeschäftigten Ausdrücke handelt es sich um eine *backward-all*-Analyse.

Ein Ausdruck ist an einer Stelle p very busy, wenn er auf allen Wegen von dort aus benutzt wird, bevor eine Redefinition eines seiner Operanden erfolgt.

Die Menge von very busy Ausdrücken am Programmende $out(B_{19})$ ist leer. Ausdrücke sind very busy am Ende eines Blocks, wenn sie am Anfang jeden Nachfolgers ebenfalls very busy sind.

Am Anfang eines Blocks B_i sind diejenigen Ausdrücke very busy, die in B_i benutzt werden ($gen(B_i)$) sowie die, die am Ende von B_i bereits very busy sind und deren Operanden innerhalb des Blocks nicht redefiniert werden ($kill(B_i)$). Es gilt also:

$$out(B_{19}) = \varnothing \qquad out(B_i) = \bigcap_{B_j \in suc(B_i)} in(B_j)$$

$$in(B_i) = gen(B_i) \cup (out(B_i) - kill(B_i))$$

Für jeden Block berechnen wir nun $gen(B_i)$ und $kill(B_i)$. Die Menge $gen(B_i)$ enthält alle Operationen X **op** Y oder **op** X aus B_i, deren Operanden X oder Y nicht vorher im Block B_i (re-) definiert worden sind. Die Menge $kill(B_i)$ enthält alle Ausdrücke X **op** Y oder **op** X aus B_i, deren Operanden X oder Y bis zum Ende des Blockes B_i nicht redefiniert werden.

Konstanten sowie einfache Speicher- bzw. Registerzugriffe betrachten wir hier nicht als Ausdrücke, da sie Zugriff ohne zusätzlichen Berechnungsaufwand darstellen. Wir erhalten die folgenden *kill*- und *gen*-Mengen:

i	$gen(B_i)$	$kill(B_i)$
1	∅	∅
2	{T2 mod 2, T1 mod 2}	{T2 mod 2, T1 mod 2}
3	∅	∅
4	{T1/2, T2/2, T3+1}	∅
5	{–T2}	{–T2}
6	∅	∅
7	∅	∅
8	{T4 mod 2}	{T4 mod 2}
9	{T4/2}	∅
10	∅	∅
11	{–T4}	{–T4}
12	∅	∅
13	{T1–T2}	{T1–T2}
14	∅	∅
15	∅	∅
16	∅	∅
17	{T2*T1, T3–1}	∅
18	∅	∅
19	∅	∅

Durch Anwendung der beschriebenen Regeln auf jeden Basisblock B_i können wir das Datenflussgleichungssystem aufstellen.

i	$out(B_i)$	$in(B_i)$
1	$in(B_2)$ = {T2 mod 2, T1 mod 2}	∅ ∪ ({T2 mod 2, T1 mod 2} – ∅) = {T2 mod 2, T1 mod 2}

i	$out(B_i)$	$in(B_i)$
2	$in(B_3) \cap in(B_6) = \varnothing$	{T2 mod 2, T1 mod 2} $\cup$ ($\varnothing$ – {T2 mod 2, T1 mod 2}) = {T2 mod 2, T1 mod 2}
3	$in(B_4) \cap in(B_5)$ = {T1/2, T2/2, T3+1, T2 mod 2, T1 mod 2} $\cap$ {-T2} = $\varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
4	$in(B_2)$ = {T2 mod 2, T1 mod 2}	{T1/2, T2/2, T3+1} $\cup$ ({T2 mod 2, T1 mod 2} – $\varnothing$) = {T1/2, T2/2, T3+1, T2 mod 2, T1 mod 2}
5	$in(B_7) = \varnothing$	{-T2} $\cup$ ($\varnothing$ – {-T2}) = {-T2}
6	$in(B_7) = \varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
7	$in(B_8) \cap in(B_{14}) = \varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
8	$in(B_9) \cap in(B_{10}) = in(B_9) \cap$ {T1-T2}	{T4 mod 2} $\cup$ ($out(B_8)$ – {T4 mod 2})
9	$in(B_8)$	{T4/2} $\cup$ ($out(B_9) - \varnothing$)
10	$in(B_{11}) \cap in(B_{12})$ = {-T4, T1-T2} $\cap$ {T1-T2} = {T1-T2}	$\varnothing \cup$ ({T1-T2} – $\varnothing$) = {T1-T2}
11	$in(B_{13})$ = {T1-T2}	{-T4} $\cup$ ({T1-T2} – {-T4}) = {-T4, T1-T2}
12	$in(B_{13})$ = {T1–T2}	$\varnothing \cup$ ({T1-T2} – $\varnothing$) = {T1-T2}
13	$in(B_7) = \varnothing$	{T1-T2} $\cup$ ($\varnothing$ – {T1–T2}) = {T1-T2}
14	$in(B_{15}) \cap in(B_{18}) = \varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
15	$in(B_{16}) = \varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
16	$in(B_{17}) \cap in(B_{19}) = \varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
17	$in(B_{16}) = \varnothing$	{T2*T1, T3-1} $\cup$ ($\varnothing - \varnothing$) = {T2*T1, T3-1}
18	$in(B_{19}) = \varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$
19	$\varnothing$	$\varnothing \cup (\varnothing - \varnothing) = \varnothing$

Nur für die Blöcke B_8 und B_9 können die *in*- und *out*-Mengen nicht direkt bestimmt werden.

Da wir eine *backward*-Analyse vornehmen, initialisieren wir zur iterativen Lösung des Gleichungssystems die $in(B_i)$ nicht mit $\varnothing$, sondern jeweils mit der Menge U aller im Programm vorkommenden Ausdrücke, U={T2 mod 2, T1 mod 2, T1/2, T2/2, T3+1, -T2, T4 mod 2, T4/2, -T4, T1-T2, T2*T1, T3-1}.

Wir setzen $in(B_i) = U$, $0 < i < 14$, und berechnen in- und out-Mengen im Graphen jeweils von hinten nach vorne (in der Tabelle von unten nach oben, wobei immer das aktuellste Ergebnis verwendet wird). Dabei müssen wir nur noch B_8 und B_9 betrachten. Ergebnis der ersten Iteration:

i	$out(B_i)$	$in(B_i)$
8	$in(B_9) \cap$ {T1-T2} = $U \cap$ {T1-T2} = {T1-T2}	{T4 mod 2} $\cup$ ($out(B_8)$ – {T4 mod 2}) = {T4 mod 2} $\cup$ ({T1-T2} – {T4 mod 2}) = {T4 mod 2, T1-T2}

i	$out(B_i)$	$in(B_i)$
9	$in(B_8)$	$\{T4/2\} \cup (out(B_9) - \emptyset)$
	$= Y$	$= \{T4/2\} \cup (U - \emptyset)$
		$= U$

Zweite Iteration:

i	$out(B_i)$	$in(B_i)$
8	$in(B_9) \cap \{T1\text{-}T2\}$	$\{T4 \bmod 2\} \cup (out(B_8) - \{T4 \bmod 2\})$
	$= \{T4/2, T4 \bmod 2, T1\text{-}T2\} \cap \{T1\text{-}T2\}$	$= \{T4 \bmod 2\} \cup (\{T1\text{-}T2\} -$
	$= \{T1\text{-}T2\}$	$\{T4 \bmod 2\})$
		$= \{T4 \bmod 2, T1\text{-}T2\}$
9	$in(B_8) = \{T4 \bmod 2, T1\text{-}T2\}$	$\{T4/2\} \cup (out(B_9) - \emptyset)$
		$= \{T4/2\} \cup (\{T4 \bmod 2, T1\text{-}T2\} - \emptyset)$
		$= \{T4/2, T4 \bmod 2, T1\text{-}T2\}$

Dritte Iteration:

i	$out(B_i)$	$in(B_i)$
8	$in(B_9) \cap \{T1\text{-}T2\}$	$\{T4 \bmod 2\} \cup (out(B_8) - \{T4 \bmod 2\})$
	$= \{T4/2, T4 \bmod 2, T1\text{-}T2\} \cap \{T1\text{-}T2\}$	$= \{T4 \bmod 2\} \cup (out(B_8) - \{T4 \bmod 2\})$
	$= \{T1\text{-}T2\}$	$= \{T4 \bmod 2, T1\text{-}T2\}$
9	$in(B_8) = \{T4 \bmod 2, T1\text{-}T2\}$	$\{T4/2\} \cup (out(B_9) - \emptyset)$
		$= \{T4/2\} \cup (\{T4 \bmod 2, T1\text{-}T2\} - \emptyset)$
		$= \{T4/2, T4 \bmod 2, T1\text{-}T2\}$

Auch jede weitere Iteration liefert dasselbe Ergebnis. Damit haben wir bereits einen Fixpunkt erreicht und das Gleichungssystem gelöst.